NASA
National Aeronautics and
Space Administration

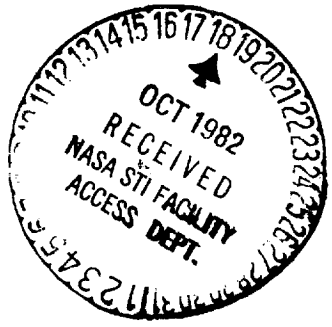
MAY 13 1982

Lyndon B. Johnson Space Center
Houston, Texas 77058

JSC-17841

UNIFIED DATA BASE MANAGEMENT SYSTEM (UDBMS)
USER'S GUIDE

Job Order 45-505



Prepared By

Lockheed Engineering and Management Services Company

Houston, Texas

Contract NAS 9-15800

For

EXPERIMENT SYSTEMS DIVISION

March 1982

LEMSCO-17354, Rev. A

JSC-17841

UNIFIED DATA BASE MANAGEMENT SYSTEM (UDBMS)
USER'S GUIDE

Job Order 45-505

PREPARED BY

C. C. de Valcourt

E. V. Brown

N. Massom-Zadeh

J. L. Allison

APPROVED BY

LOCKHEED	NASA/JSC
J. E. Vinson, Supervisor Data Processing & Analysis Section	R. F. Irwin, Head Computer Systems Section
B. O. Lippert, Manager Experiments Systems Department	J. H. Langford, Chief Systems Engineering Branch

Prepared For

Experiment Systems Division

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
LYNDON B. JOHNSON SPACE CENTER
HOUSTON, TEXAS

March 1982

LEMSCO-17354,
Rev. A

TECHNICAL REPORT INDEX/ABSTRACT
(See instructions on reverse side.)

1. TITLE AND SUBTITLE OF DOCUMENT
Unified Data Base Management System (UDBMS) User's Guide

2. JSC NO.
JSC-17841

3. CONTRACTOR/ORGANIZATION NAME
Lockheed Engineering & Management Services Company

4. CONTRACT OR GRANT NO.
NAS 9-15800

5. CONTRACTOR/ORIGINATOR DOCUMENT NO.
LEMSCO-17354, Rev. A

6. PUBLICATION DATE (THIS ISSUE)
March 1982

7. SECURITY CLASSIFICATION
Unclassified

8. OPR (OFFICE OF PRIMARY RESPONSIBILITY)
JSC - Experiment Systems Division

9. LIMITATIONS
GOVERNMENT HAS UNLIMITED RIGHTS [X] YES [ ] NO

IF NO, STATE LIMITATIONS AND AUTHORITY

10. AUTHOR(S)
C. C. de Valcourt
E. V. Brown
N. Massom-Zadeh
J. L. Allison

11. DOCUMENT CONTRACT REFERENCES

WORK BREAKDOWN STRUCTURE NO.
Job Order 45-505

CONTRACT EXHIBIT NO.

DRL NO. AND REVISION

DRL LINE ITEM NO.

12. HARDWARE CONFIGURATION

SYSTEM Unified Data Base Management System (UDBMS) User's Guide

SUBSYSTEM

MAJOR EQUIPMENT GROUP
Hewlett-Packard 3000/Series III, (MPE-III

13. ABSTRACT

This document provides a complete guide on using the Unified Data Base Management System (UDBMS) for Data Base Manager's (DBM), Data Base Administrators (DBA), and any other user attempting to update or retrieve and report information from a UDBMS data base.

The document includes an introduction, a system overview, background notes about the development of UDBMS, and a summary of major system features. The main body of the document is an alphabetical listing of the simple (2 character) commands, syntax to enter variations of the commands, and one or more examples to illustrate some command usage. Several appendices are included to provide more concise listings of commands.

14. SUBJECT TERMS

Unified Data Base Management System
UDBMS
Data Base Management Systems
DBMS
Management Information Systems
Data Base Management
USING UDBMS
Data Base Administration

LIST OF EFFECTIVE PAGES

The list of effective pages gives the date of the current edition and of any pages changed in updates to that edition. Within the manual, any pages changed since the last edition are indicated by printing the revision number in the upper right hand corner.

First Edition.....November 1981
Revision 1 (Update).....March 1982

SECTION	PAGE(S)	PREV.	CHANGE	
Title page	i	OI	R1*	March 82
Signature Page	iii	OI	R1*	March 82
Abstract	v	OI	R1*	March 82
Contents	ix	OI	R1*	March 82
	x	OI	R1*	March 82
Section 2	2 - 7	OI	R1*	March 82
	2 - 8	OI	R1*	March 82
Section 3	3 - 147	OI	R1*	March 82
	3 - 147a-c	--	R1**	March 82
	3 - 148	OI	R1*	March 82
	3 - 202a-c	--	R1**	March 82
Appendix A	App. A	OI	R1*	March 82
Appendix B	App. B	OI	R1*	March 82
Appendix F	App. F	OI	R1*	March 82

LEGEND: * Replace Page(s) ** New Page(s)
OI - Original Edition
R1 - Revision A

CONTENTS (Cont.)

CONTENTS (Cont.)

APPENDICES

LIST OF FIGURES

2.5.1 APP-Application Subsystem

The UDBMS application subsystem contains all commands which are data base dependent;ie..., they are usually useful only to those users on a particular data base and have been developed to suit the needs of users who could not accomplish specific data base requirements using the generalized UDBMS subsystems. The APP commands for the RIMS and REDAF data bases are: DP, SG, TL, and UG.

2.5.2 DDL-Data Definition Language

These are the commands which create, locate, select, display, and remove Data Definition format records in a given data base. The DDL commands are: DD, FO, IF, LF, and RF.

2.5.3 DML-Data Manipulation Language

The DML commands are update commands that actually change data records and/or keys in a data base. These commands may affect the records, the Key Index, or both. The DML commands are: AF, AK, AN, AR, CF, CS, DK, DS, ES, KY, NK, PO, QE, RE, RK, RR, RU, UP, VP, and ZK.

2.5.4 GUL-General User Language

The GUL commands are those which are standard operating commands and which may be related to some of the other subsystems. The GUL commands are: BE, DI, EX, HE, QL, RD, RV, ST, TO, and XR.

2.5.5 MGR-Manager

The MGR commands are those which are used by the Data Base Manager involved in mass updates, field restructuring, security, base verification, etc. The MGR commands are: AU, DC, K0, K1, K2, K3, PA, RE, S+, S-, SM, and VS.

2.5.6 MISC-Miscellaneous

Other MISC commands not easily classified are: <<, >>, AD, CR, D0, D1, IB, ME, MO, XF, XS, and cY.

2.5.7 QRL-Query/Report Language

The QRL commands are those commands which are used to select sets, select output devices and/or report characteristics, and finally report the data. The QRL commands are : B0, B1, C0, C1, CC, CL, DA, DF, DJ, DT, DU, EP, ER, FA, FC, FD, FT, HD, ID, IP, LS, MF, OS, PS, SC, SD, SF, SH, SJ, SK, SL, SR, SS, SU, UF, UN, and XK.

2.5.8 SML-Set Manipulation Language

The SML commands are commands which are used to refine data sets in the users temporary status table. These commands allow sorting, combinations, etc., of data prior to reporting or updating. The SML commands are: CO, CP, DE, IS, RP, SI, SN, SO, and SP.

2.6 USER PHASES

Generally speaking, all users will go through a similar User Phase sequence. A user will first log onto a data base of interest using the BEgin command. The user may then go into a Browse Phase where the Key Index can be viewed using the EXpand command and eventually advance into a Selection Phase where data selection begins.

Once the general data selection is complete, most users will proceed to a Refinement phase where they COmbine, SPlit, and/or SOrt the temporary data sets in the status table. At this point the user may Report and/or Update. A summary of the user operational phases and some of the associated subsystems are shown below:

- o BROWSE & SELECT (MISC, DDL, GUL)
- o REFINE (SML)
- o SETUP & REPORT (QRL)
- o UPDATE (DML, DDL, MGR)

Refer to Appendix B for a complete listing of the UDBMS commands by Phase and Subsystem.

2.7 USER TYPES

It is important to discuss the general types of data base users in the average data processing environment. The first type of user discussed is the Read-only user who is interested in accessing existing data bases. The Read-only user will be most interested in Selection, Refinement, Set-up, and Report commands and is not interested in update commands or manager level functions on a data base.

On the other hand, an Update user will be interested in all of the above as well as the Update commands. For reference purposes, two sample sessions of users running UDBMS can be found in Appendix E. The first user is a turnkey (Read- only) user and the other is a Manager (Update) user who creates a data base, initializes it, updates, selects, and reports.

RR

command purpose REPLACE RECORD

command usage MANAGER LEVEL SYSTEM CHECKOUT

command syntax RRrecid,length,text

other comments THIS COMMAND IS NOT A STANDARD USER COMMAND

special notes THIS COMMAND IS USED DURING SYSTEM CHECK

command result recid IS REPLACED BY text and NEW length USED

system response IF SUCCESSFUL-NO SYSTEM RESPONSE

last update FRI, OCT 9, 1981, 9:50 AM

error messages RECORD NOT FOUND

update?(Y/N) Y user level M dialog?(Y/N) N complexity M

effect of crash on base? LDB software subsystem DML

user operational phase CHECKOUT COMMAND MASK# 57

NOTE: - This command is usually not needed or used by standard users. It is, however, a very important system checkout command during installation.

The RR command replaces the existing text of the designated record with the input text.

EXAMPLE:- 10:15A COMMAND?RR1001,20,THIS IS A REPLACEMENT TEST.

EXPLANATION:- The UDBMS installer has entered at least one record and wants to make a quick check. In this instance, record 1001 has been replaced by a new record with a length of 20 (80 characters) and the text: THIS IS A REPLACEMENT TEST.

RU

command purpose RPN UPDATE

command usage COMPLEX UPDATES

command syntax RUset#

other comments MUST BE PRECEEDED BY AN SC COMMAND

special notes ALLOWS .STO. INTO ANY OPERAND

command result RECORDS ARE UPDATED AS SPECIFIED

system response RECORD XXXX UPDATED

last update MON, JAN 25, 1982, 2:09 PM

error messages RU MUST BE PRECEEDED BY AN SC

update?(Y/N) Y user level G dialog?(Y/N) N complexity H

effect of crash on base? MLDB software subsystem DML

user operational phase UPDATE COMMAND MASK# 85

NOTE: Additional RPN operators are available for use with the RU command. See Appendix F for an updated list of valid operators.

In this example, three payroll records (#FMT559) are selected and updated using the RPN Update command. Note that the RU command must be preceeded by at least one Set Calculation (SC) command which tells UDBMS what you want to do.

```
Select the data set to be updated.....
 9:10A COMMAND?SK#FMT559
        1   #FMT559                          3

Below is the format of the records to be updated......
 9:10A COMMAND?FO559
 FIXED FORMAT
 NCTL=  0   NTXT=  0
              *** FORMAT 559 *** DATA BASE K1.PUB
                LENGTH  51                    NO. FLDS    8
FLD NO     FIELD NAME     START(CHAR)   LENGTH(CHARS)   TYPE  KEY
 1  NAME                        1             20          1    1
 2  DEPARTMENT                 21             12          1    1
 3  PAY RATE                   33              8          2    1
 4  TOTAL HOURS                41              6          2    1
 5  HRS TO DATE                47             10          2    0
 6  TOTAL PAY                  57             12          2    0
```

```
 7  PAY TO DATE                69          20          2   0
 8  EMPLOYEE NUMBER            89          20          2   0

Issue some quick look (QL) commands to see the data......
 9:11A COMMAND?QL1,NAME,PAY RATE,TOTAL HOURS
JONES, JIM              10.22     31
SMITH, TOMMY             9.92     40
CHAN, CHARLIE           11.42     40

9:11A COMMAND?QL1,NAME,PAY RATE,TOTAL HOURS,TOTAL PAY,HRS TO DATE
JONES, JIM              10.22     31    12611.50        1234
SMITH, TOMMY             9.92     40    12201.60        1230
CHAN, CHARLIE           11.42     40    14160.80        1240

Issue Set Calculations commands to set up update conditions
desired. SC1 specifies that PAY RATE should be multiplied by the
TOTAL HOURS and the result stored into the TOTAL PAY field. SC2
adds TOTAL PAY to PAY TO DATE and stores the result back into PAY
TO DATE. SC3 adds TOTAL HOURS to HRS TO DATE and stores the
result back to HRS TO DATE.

 9:12A COMMAND?SC1,PAY RATE,TOTAL HOURS,.MLT.,TOTAL PAY,.STO.

 9:13A COMMAND?SC2,TOTAL PAY,PAY TO DATE,.ADD.,PAY TO DATE,.STO.

 9:13A COMMAND?SC3,HRS TO DATE,TOTAL HOURS,.ADD.,HRS TO
 DATE,.STO.

Now issue the RU command to perform the updates specified above..
 9:14A COMMAND?RU1

**RECORD:    6331 UPDATED  **
**RECORD:    6332 UPDATED  **
**RECORD:    6333 UPDATED  **
 9:14A COMMAND?QL1,NAME,TOTAL PAY,PAY TO DATE,HRS TO DATE
JONES, JIM              316.82          12928.30    1265.00
SMITH, TOMMY            396.80          12598.40    1270.00
CHAN, CHARLIE           456.80          14617.60    1280.00

Now dump the entire set using Display Tabular......
 9:15A COMMAND?DT1
MON, MAR 15, 1982,  9:15 AM ON DEV 49                  PAGE    1


                BASE K1.PUB  FMT#559  #REC 3  SET #FMT559


NAME                JONES, JIM
DEPARTMENT          RESOURCES
PAY RATE               10.22
TOTAL HOURS            31
HRS TO DATE          1265.00
TOTAL PAY               316.82
```

PAY TO DATE	12928.30
EMPLOYEE NUMBER	99-98-77
NAME	SMITH, TOMMY
DEPARTMENT	RESOURCES
PAY RATE	9.92
TOTAL HOURS	40
HRS TO DATE	1270.00
TOTAL PAY	396.80
PAY TO DATE	12598.40
EMPLOYEE NUMBER	44-55-99
NAME	CHAN, CHARLIE
DEPARTMENT	PAYROLL
PAY RATE	11.42
TOTAL HOURS	40
HRS TO DATE	1280.00
TOTAL PAY	456.80
PAY TO DATE	14617.60
EMPLOYEE NUMBER	33-44-77

(INTENTIONALLY BLANK)

RV

command purpose REVERT BASE TO STATUS AT THE LAST BE COMMAND

command usage UPDATE RECOVERY

command syntax RV

other comments USE VERY SPARINGLY; AS A LAST RESORT

special notes

command result BASE REVERTS TO ITS STATUS BEFORE THE LAST BE

system response ##**PAGES WRITTEN BACK TO BASE**

last update TUE, JUN 30, 1981, 10:50 AM

error messages NONE

update?(Y/N) Y user level G dialog?(Y/N) N complexity L

effect of crash on base? LDB software subsystem GUL

user operational phase UPDATE COMMAND MASK# 69

EXAMPLE:- 10:15A COMMAND?RV (INPUT)

1122 **PAGES WRITTEN BACK INTO BASE** (RESPONSE)

EXPLANATION:- The user decided that early in a lengthy update session an entry error occurred which would grossly foul the data base. The decision was made to revert the data base to the last BE command in order to recover. In other words, the user decided to "bail out" and reenter the proper update information rather than try to salvage.

XF

command purpose EXECUTE COMMANDS IN PROCEDURE FILES

command usage REPORTS,DATA SELECTION,CANNED PROCEDURES

command syntax XF (prompts for file name)

other comments READ GROUND RULES BELOW BEFORE USING

special notes CONTROL Y STOPS THE EXECUTION

command result COMMANDS EXECUTED ACCORDING TO LOGIC IN FILE

system response DEPENDS ON EXECUTION FILE LOGIC

last update THU, FEB 18, 1982, 11:00 AM

error messages INVALID SYNTAX/INVALID FILE

update?(Y/N) N user level G dialog?(Y/N) N complexity M

effect of crash on base? MLDB software subsystem MISC

user operational phase ANY COMMAND MASK#

This module executes a special procedure file .The file may contain any valid UDBMS command as well as control lines. All control lines must be preceded with a ! symbol in column 5 and allow conditional branching between execution of the UDBMS commands in this file. The XF procedure file has a special format. Columns 1-4 are assigned for statement labels which are the numeric digits 1-900 similar to FORTRAN statement labels and are used for the GOTO statements described below. All UDBMS commands and control lines must begin in any column 5 thru 10. The control lines have a special syntax and allow testing of data selection status as well as boolean Reverse Polish Notation (RPN) conditions. The syntax for control lines is outlined below:

! <control statement> ; action statement

Where the optional control statement is defined as:

1. HIT,0--meaning if there were no hits on the previous selection or combine.
   OR
2. HIT,1--meaning if there were hits on the previous selection or combine.
   OR
3. any valid boolean RPN condition which can be evaluated as TRUE or FALSE.
   (i.e. !PROJ NO,'0547R3',.EQ.;GOTO10)

NOTE: In order to utilize the HIT check, the check must immediately follow the selection or combination statement in the procedure file.

The action statement is defined as:

```
GOTO XXX
   or
EXIT
```

AN EXAMPLE XF file is shown below:

```
1      BE              <<CLEAR STATUS TABLE>>
       SKDEPARTMENT:(*PLEASE INPUT DEPARTMENT*)     (USER PROMPT)
      !HIT,0;EXIT     <<exit file when there are no hits>>
       SO1,NAME        <<sort set on name                 >>
       DT1             <<display tabular the set          >>
      !GOTO 1          <<do it again                      >>
```

Execution of this file is shown below:

```
 9:59A COMMAND?XF

PROCEDURE FILE NAME ?XFILE

PLEASE INPUT DEPARTMENT
RESOURCES
        1   DEPARTMENT:RESOURCES              2

TUE, MAR 16, 1982, 10:00 AM ON DEV 49              PAGE 1


   BASE K1.PUB  FMT#559  #REC 2  SET DEPARTMENT:RESOURCES


NAME                  JONES, JIM
DEPARTMENT            RESOURCES
PAY RATE                 10.22
TOTAL HOURS               31
HRS TO DATE             1265.00
TOTAL PAY                316.82
PAY TO DATE                    12928.30
EMPLOYEE NUMBER                 99-98-77

NAME                  SMITH, TOMMY
DEPARTMENT            RESOURCES
PAY RATE                  9.92
TOTAL HOURS               40
HRS TO DATE             1270.00
TOTAL PAY                396.80
PAY TO DATE                    12598.40
EMPLOYEE NUMBER                 44-55-99
```

```
PLEASE INPUT DEPARTMENT
SECURITY
        1    DEPARTMENT:SECURITY                1
TUE, MAR 16, 1982, 10:00 AM ON DEV 49             PAGE 1

   BASE K1.PUB  FMT#2  #REC 1  SET DEPARTMENT:SECURITY


DEPARTMENT            SECURITY
NO OF PERSONEL           8
HOURS WORKED             205.4
AVERAGE RATE              8.87

PLEASE INPUT DEPARTMENT


NO HIT-NO SET FORMED   (UERR 22)

XFILE            PROCEDURE FILE EXECUTION COMPLETED
```

---COMMAND INDEX---

CMD	LE-VEL	UPD-ATE	SYNTAX	PURPOSE
:	G	N	:	TERMINATE DATA BASE SESSION
<<	G	N	<<comment	ADD FILE OR XK COMMENT LINE
>>	G	N	>>text	ECHO TEXT TO USER
AD	G	N	ADfilename<.group.account>	ADD COMMANDS FROM FILE
AF	G	Y	AF	ADD RECORDS FROM FILE
AK	M	Y	AKkeyname	ADD KEY TO DATA BASE
AN	M	Y	AN	ADD RECORDS FROM FILE
AR	G	Y	AR<recid,length,text>	ADD RECORD TO DATA BASE
AU	M	Y	AU	AUTOMATIC UPDATE SUBSYSTEM
B0	G	N	B0	BLANKS ON
B1	G	N	B1	BURN TRAILING BLANKS
BE	G	N	BE<data base name>	BEGIN & INITIALIZE
C0	G	N	C0	CALCULATIONS OFF
C1	G	N	C1	CALCULTIONS ON
CC	G	N	CCcc#,bool.;start#@stop#	CALCULATION CONTROL
CF	G	Y	CFset#,field=value<,...>	CHANGE FIELD
CL	G	N	CLreg#@reg#	CLEAR DOLLAR REGISTERS
CO	G	N	COset#.OP.set#	COMBINE SETS (BOOLEAN)
CP	G	N	CPwff	REVERSE POLISH COMBINE
CR	M	N	CR<program.group.acct>	CREATE PROCESS(RUN PROGRAM)
CS	G	Y	CSset#	COPY A SET
D0	G	N	D0	TURN OFF DBUG
D1	G	N	D1	TURN ON DBUG MODE
DA	G	N	DAset#,fmt#,<not sure>	DISPLAY AND SUM
DC	M	N	DC	DATA DICTIONARY SUBSYSTEM
DD	G	Y	DDfmt#,reclen	DATA DEFINITION
DE	G	N	DEset#	DELETE ENTRY
DF	G	N	DFset#,fmt#<'title'>	DISPLAY FORMATTED
DI	G	N	DIset#<,start#,stop#>	DISPLAY INTERNAL RECORD
DJ	G	N	DJset#,fmt#	DISPLAY JOINT
DK	M	Y	DKkeyname	DELETE KEY
DP	G	Y	DPform,set#<,[Y][N]	DPAR COMMAND
DS	G	Y	DSset#	DELETE SET
DT	G	N	DTset#<,fmt#>	DISPLAY TABULAR
DU	G	N	DU[1][0]	DUPLICATE CONTROL
EP	G	N	EP	EJECT PAGE AT OUTDEV
ER	G	N	ER	EDIT RPN(CLEAR DOLLARS)
ES	G	Y	ESset#,fmt#	EDIT SET
EX	G	N	EX<keyname><,exsize>	EXPAND
FA	G	N	FAset#,field<,field..>	FIELD ACCUMULATE
FC	G	N	FCfilename,set#	FORM CONDENSE
FD	G	N	FD<filename,set#>	FORM DISPLAY
FO	G	N	FOfmt#	DISPLAY A FORMAT(DATA DEF.)
FT	G	N	FTfilename<,set#>	FOOT REPORT
HD	G	N	HDfilename.group.account	TURN ON HEADER FROM FILE
HE	G	N	HE,command	HELP SUBSYSTEM

---COMMAND INDEX---

CMD	LEVEL	UPDATE	SYNTAX	PURPOSE
IB	G	N	IB	IDENTIFY BASE-(CURRENT)
ID	G	N	ID	INITITALIZE DOLLARS
IF	G	N	IF	IDENTIFY FORMATS
IP	G	N	IP	INCREMENT PAGE NUMBER BY 1
IS	G	N	ISset#	INVERT SET
K0	M	Y	K0fmt#,fieldname	UNKEY A FIELD TO KEY TYPE 0
K1	M	Y	K1fmt#,fieldname	KEY A FIELD TO KEY TYPE 1
K2	M	Y	K2fmt#,fieldname	KEY A FIELD TO KEY TYPE 2
K3	M	Y	K3fmt#,fieldname	KEY A FIELD TO KEY TYPE 3
KY	M	Y	KYset#,fmt#	KEY A SET
LF	G	N	LFfieldname	LOCATE FIELD
LS	G	N	LSnlpp	LINE SELECT
ME	G	N	ME,any mpe command	MPE LINK
MF	G	N	MF	MERGED FORM DISPLAY
MO	G	N	MO	MONITOR ON
NK	M	Y	NKset#,fmt#	NON-KEY A SET
OS	G	N	OS#	OVERSTRIKE OUTPUT
PA	M	Y	PA	PACK (INTERACTIVE)
PO	M	Y	POsetA,setB	POST
PS	G	N	PSpage#	PAGE SELECT
QE	G	Y	QEset#,fld<,fld,fld,...>	QUICK EDIT
QL	G	N	QLset#,field,field,...	QUICK LOOK
RD	G	N	RD	REDO LAST COMMAND
RE	M	Y	REset#,newfmt#	RESTRUCTURE RECORDS
RF	M	Y	RFfmt#	REMOVE FORMAT
RK	M	Y	RKoldkeyname@newkeyname	REPLACE KEY
RP	G	N	RPset#,wff	REVERSE POLISH SELECT
RR	M	Y	RRrecid,length,text	REPLACE RECORD
RU	H	Y	RUset#	RPN UPDATING
RV	G	Y	RV	REVERT BASE TO LAST BE
S+	M	Y	S+	ADD PASSWORD TO BASE
S-	M	Y	S-	DELETE DATA BASE PASSWORD
SC	G	N	SC[cmd#,wff][keyname]	SET CALCULATIONS(RPN)
SD	G	N	SD<device#>	SET OR SELECT OUTPUT DEVICE
SF	M	N	SFset#,fmt#	SPOOL FORMATTED
SG	G	N	SG	SELECT GEOREF
SH	G	N	SH<header option>	SELECT HEADER
SI	G	N	SIset#,field<,field,....>	INDIRECT SELECT
SJ	M	N	SJset#,fmt#	SPOOL JOINT
SK	G	N	SKkeyname<@keyname>	SELECT BY KEY
SL	G	N	SL#@#	SET LIMIT(CLEAR DOLARS)
SM	M	N	SMmode#,value	SET MODE
SN	G	N	SNset#,field.OP.value<,..>	SELECT NON-KEYED
SO	G	N	SOset#<,field,field,...>	SORT

---COMMAND INDEX---

CMD	LE-VEL	UPD-ATE	SYNTAX	PURPOSE
SP	G	N	SPset#	SPLIT SETS BY FORMAT
SR	G	N	SRrecid	SELECT RECORD(REC. NUMBER)
SS	G	N	SS<set title>	SPECIFY SET
ST	G	N	ST<start number>	STATUS (STATUS TABLE)
SU	G	N	SU<fmt#>	SELECT UNIVERSE
TL	G	N	TL	TAPE LIST(ARCHIVE REPORT)
TO	M	N	TO	TURN OFF SYSTEM TIMER
UF	G	N	UF	UNDERLINE OFF(FORM DISPLAY)
UG	M	Y	UGset#	UPDATE GEOREF
UN	G	N	UN	UNDERLINE ON
UP	M	Y	UPsetA,setB	UNPOST
VP	M	Y	VPset#,set#	VERIFY POST
VS	M	N	VS	VERIFY SETS
XF	M	N	XF	EXECUTE PROCEDURE FILE
XK	G	N	XKkeyname	EXECUTE KEY
XR	G	N	XRset#	CROSS REFERENCE A SET
XS	G	N	XSset#	EXECUTE SET
ZK	M	Y	ZKkey#	ZAP KEY FROM EXPAND TABLE
cY	G	N	cY	SUBSYSTEM BREAK

APPENDIX B

COMMANDS BY SUBSYSTEM & USER PHASE

FRI, MAR 19, 1982

COMMANDS BROKEN DOWN BY SUBSYSTEM AND USER PHASE
(SUBSYSTEM,PHASE)

****

(APP ,REPORT)

DP,SG,TL

(APP ,UPDATE)

UG

(DDL ,ANY)

FO

(DDL ,SELECT)

IF,LF

(DDL ,UPDATE)

DD,RF

(DML ,CHECKOUT)

AK,AR,DK,PO,RK,RR,UP,VP

(DML ,UPDATE)

AF,AN,CF,CS,DS,ES,KY,NK,QE,RE,RU,ZK

(GUL ,ANY)

:,BE,EX,HE,RD,ST,TO,XR

(GUL ,REPORT)

DI,QL

(GUL ,UPDATE)

RV

(MGR ,ANY)

DC,SM,VS

(MGR ,UPDATE)

AU,K0,K1,K2,K3,PA,S+,S-

(MISC ,ANY)

<<,>>,AD,CR,D0,D1,IB,ME,MO,XF,XS,cY

(QRL ,ANY)

ER,SF,SJ

(QRL ,REPORT)

CC,DA,DF,DJ,DT,EP,FA,FC,FD,FT,HD,ID,IP,MF,SD,XK

(QRL ,SELECT)

SK,SR,SS,SU

FRI, MAR 19, 1982

COMMANDS BROKEN DOWN BY SUBSYSTEM AND USER PHASE
(SUBSYSTEM,PHASE)

****

(QRL ,SETUP)
B0,B1,C0,C1,CL,DU,LS,OS,PS,SC,SH,SL,UF,UN

(SML ,REFINEMENT)
CO,CP,DE,IS,SO,SP

(SML ,SELECT)
RP,SI,SN

THU, NOV 12, 1981 9:44 AM EVB EVB SDUDBMS

TOTAL NUMBER OF UPDATE COMMANDS : 33
TOTAL NUMBER OF READ/ONLY COMMANDS : 76

TOTAL NUMBER OF COMMANDS : 109

REVERSE POLISH NOTATION (RPN)

In 1951 the Polish logician Jan Lukasiewicz devised parenthesis-free notation for logic. This method of writing an expression is referred to as 'polish notation' in honor of Lukasiewicz and has been extended for use in algebra and other operator-operand systems.

A valid polish notation expression is formed basically by consistently placing operators before (Pre- fix) or after (Post-fix) their operands; the need for parentheses is eliminated provided each operator has a fixed number of operands. The RPN (Post-fix polish form) is used as an intermediate expression form for many language compilers. In the case of evaluating a Post-fix Reverse Polish Notation expression restricted to only binary operators, the number of operators will always be equal to the number of operands-1. When evaluating a unary RPN expression, the number of operators will be equal to the number of operands. The UDBMS RPN processor allows mixed unary and binary expressions in a single string which is referred to as a Well Formed Formula (WFF).

UDBMS RPN COMMAND PROCESSING

UDBMS commands are available which allow stacked boolean and arithmetic conditions to be specified for use in data selection, calculated reports, and set combinations. These commands are:

1. RP- reverse polish select allows the user to select records based on RPN conditions specified by a user.

2. SC- set calculations set up RPN report specifications prior to processing.

3. CC- calculation control for conditional calculation control.

4. SL- set limit on break point clearing of registers.

5. CP- combine sets using RPN syntax.

The UDBMS RPN processor can evaluate these WFF's in regards to several types of operands which are detailed below.

VALID RPN OPERANDS

FIELD NAME- any valid data base field may be used as a valid operand as long as it is specified exactly as it is described in a given data base format.

FIELD NUMBER- the relative field number within a data base format may be used instead of the complete field name.

(START CHARACTER:NO OF CHARACTERS)- this type of operand is the absolute method of specifying any portion of a data record to be used as an operand.

'ANY CHARACTER STRING'- this operand format allows the usage of character and numeric string constants to be included in WFF's.

$N- any one of the RPN $ scratch registers 1 through 50 can be used as a valid operand for calculations.

RPN BINARY OPERATORS

.EQ.,.NE.	-equal to, not equal to
.GT.,.LT.	-greater than, less than
.GE.,.LE.	-greater than or equal to, less than or equal to
.NDX.	-scan first operand looking for any occurence of the second.
.AND.,.OR.,.XOR.	-logical AND, logical OR, exclusive OR
.FMT.	-pop fortran format, pop operand, reformat value using format and push result back onto stack.

RPN UNARY OPERATORS

.NOT.	-negate the top of the stack.
.DUP.	-duplicate top of stack and push duplicate
.POP.	-pop top of stack

SPECIAL CASE OPERATORS

.STO.	-pop address, then store Top of Stack(TOS) in address specified in TOS-1.
.TST.	-test TOS, clear the stack, and if .TRUE. continue processing RPN string; else stop processing.
.PRT.	-print top of stack when encountered, note that .PRT. does not change stack.

RPN EXPRESSION EVALUATION

The formal definition for a UDBMS WFF is:

(1) AN OPERAND
or
(2) A WFF FOLLOWED BY AN UNARY OPERATOR
or
(3) 2 WFFS FOLLOWED BY AN BINARY OPERATOR

To process a reverse polish command string, scan from left to right. If the next item is an operand, push it onto the stack. If the next is an binary operator , pop two off of the stack, perform the operation and push the result back on. The exceptions are the three unary operators (.NOT.,.DUP.,.POP.) and the special case operators (.PRT.,.TST., and .STO.).

FOR EXAMPLE

```
$1,COST,.ADD.,$1,.STO.    [this example will accumulate ]
(----WFF----)             [cost field in $1.            ]
(---------WFF--------)
```

A post-fix expression is evaluated from left to right as follows:

(1) When a operand is encountered, it is pushed onto the stack.

(2) When a binary operator is encountered, two values are popped from the stack, operated on, and the result is pushed back onto the stack.

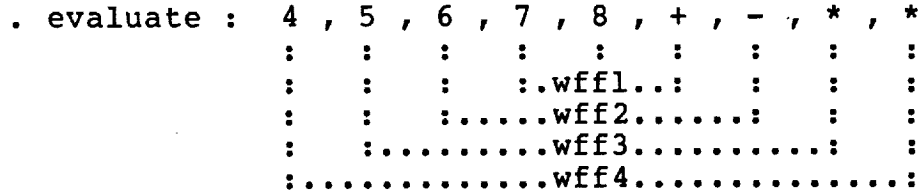
(3) When a unary operator is encountered, one value is popped from the stack, operated upon, and the result is pushed back on the stack.

Note that each set of two operands and operator forms a unit which has been referenced to as a `Well Formed Formula` or WFF. Each WFF or operand/operator set when evaluated becomes a simple operand, thus giving a recursive definition as illustrated below. The following illustrates a recursive pattern of WFF`s which when evaluated from left to right forms a single WFF-note that this is a recursive definition.

```
operand,operand,operator,operand,operator,operand,operator
:                     :                 :                 :
:.......WFF1..........:                 :                 :
        :                               :                 :
        :.........WFF2.................:                 :
                  :                                       :
                  :...............WFF3.....................:
```

o Test Cases

1). evaluate : 4 , 5 , 6 , 7 , 8 , + , - , * , *



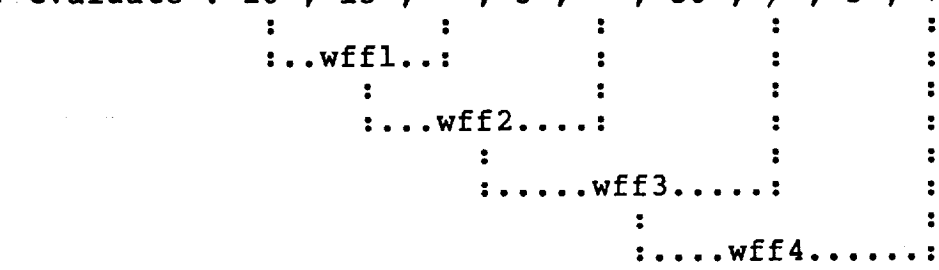
equivalent standard algebraic equation:

(((7+8)-6)*9)*4

predicted result: 180 should be value remaining on stack.

stack operations:
PUSH 4
PUSH 5
PUSH 6
PUSH 7
PUSH 8
POP 8,POP 7,ADD,PUSH 15 evaluate wff1
POP 15,POP 6,SUBTRACT,PUSH 9..evaluate wff2
POP 9,POP 5,MULTIPLY,PUSH 45..evaluate wff3
POP 45,POP 4,MULTIPLY,PUSH 180..evaluate wff4

2). evaluate : 10 , 15 , - , 3 , * , 30 , / , 3 , +



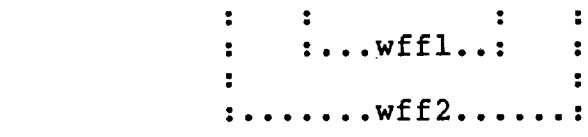
equivalent standard algebraic representation:

(30/(15-10)*3))+3

predicted result: value 5 should be value remaining on stack.

stack operations:
PUSH 10
PUSH 15
POP 15,POP 10,SUBTRACT,PUSH 5
PUSH 3
POP 3,POP 5,MULTIPLY,PUSH 15
PUSH 30
POP 30,POP 15,DIVIDE,PUSH 2
PUSH 3
POP 3,POP 2,ADD,PUSH 5

3). evaluate A , B , 103 , + , +



equivalent standard algebraic representation:

(A+(B+103))

predicted result: value 124 should be value remaining on stack.

stack operations:
PUSH A(10)
PUSH B(11)
PUSH 103
POP 103,POP B(11),PUSH 114
POP 114,POP A(10),PUSH 124

Below is a chart summarizing the RPN operators and the stack operations performed. For example, the .DIV. operator operates on two NUMeric operands, the result is also NUMeric, two arguments are popped FROM the stack, and one argument is pushed TO the stack.

RPN OPERATOR SUMMARY

OP	PURPOSE	ARG1	ARG2	RESULT	FROM	TO
.ADD.	ADD	NUM	NUM	NUM	2	1
.AND.	LOGICAL AND	BOOL	BOOL	BOOL	2	1
.DIV.	DIVIDE	NUM	NUM	NUM	2	1
.DUP.	DUPLICATE TOP OF STACK	ANY	-	ANY	0	1
.EQ.	EQUAL	ASCII	ASCII	BOOL	2	1
#EQ#	EQUAL	NUM	NUM	BOOL	2	1
.FMT.	FORMAT TOP OF STACK	*ASCII	NUM	*NUM	2	1
.GE.	GREATER THAN OR EQUAL	ASCII	ASCII	BOOL	2	1
#GE#	GREATER THAN OR EQUAL	NUM	NUM	BOOL	2	1
.GT.	GREATER THAN	ASCII	ASCII	BOOL	2	1
#GT#	GREATER THAN	NUM	NUM	BOOL	2	1
.LE.	LESS THAN OR EQUAL	ASCII	ASCII	BOOL	2	1
#LE#	LESS THAN OR EQUAL	NUM	NUM	BOOL	2	1
.LT.	LESS THAN	ASCII	ASCII	BOOL	2	1
#LT#	LESS THAN	NUM	NUM	BOOL	2	1
.MLT.	MULTIPLY	NUM	NUM	NUM	2	1
.NDX.	SEARCH FOR ARG2 IN ARG1	ASCII	ASCII	BOOL	2	1
.NE.	NOT EQUAL	ASCII	ASCII	BOOL	2	1
#NE#	NOT EQUAL	NUM	NUM	BOOL	2	1
.NOT.	LOGICAL NOT	BOOL	-	BOOL	1	1
.OR.	LOGICAL OR	BOOL	BOOL	BOOL	2	1
.POP.	POP TOP OF STACK	ANY	-	-	1	0
.PRT.	PRINT TOP OF STACK	-	-	-	0	0
.STO.	STORE	ANY	*ANY	-	1	0
.SUB.	SUBTRACT	NUM	NUM	NUM	2	1
.TST.	TEST TOP OF STACK	BOOL	-	-	*	0
.XOR.	LOGICAL EXCLUSIVE OR	BOOL	BOOL	BOOL	2	1

*ANY -this excludes .STO. operations back into a string literal.

* The .TST. operator tests, clears the stack, if .TRUE., the RPN scan continues.

*ASCII-is any valid FORTRAN format ,e.g., (F8.2), (I5),..

*NUM -is the reformatted numeric operand is above format.